HITACHI
Inspire the Next

Fixed Width Data in Pentaho Data Integration (PDI)

HITACHI
Inspire the Next

Change log (if you want to use it):

Date	Version	Author	Changes

Contents

This page intentionally left blank.

Overview

Pentaho Data Integration (PDI) offers the Fixed File Input step for reading fixed-width text files. On the output side, there is no step dedicated to this specific purpose, but fixed-width text can still be written using the existing Text file output step. This document walks you through the changes you will need to make to the default column metadata to successfully accomplish this task.

Topics covered in this document are as follows:

- Saving Fixed-Width Data
- Reading Fixed-Width Data

Our intended audience includes data analysts and ETL developers who need to write fixed-width data.

The intention of this document is to speak about topics generally; however, these are the specific versions covered here:

Software	Version(s)
Pentaho	6.x, 7.x, 8.0

The Components Reference in Pentaho Documentation has a complete list of supported software and hardware.

Before You Begin

Use the following information to prepare for the procedures described in the main section of the document.

Prerequisites

This document assumes that you have knowledge about Pentaho and programming concepts, and that you have already installed Pentaho.

Use Cases

Here are a couple of use cases that tie into this document:

Use Case 1: Legacy System Data Input

Janice, a Pentaho administrator, needs to load data into a legacy system that requires a fixed-width format dataset.

Use Case 2: Speed-Reading

Janice has a performance-critical file-loading application she is using where the file-reading speed is more important than the dataset structure. A fixed-width dataset should suit this scenario.

Saving Fixed-Width Data

Steps for saving fixed-width data using the Text file output step are detailed in the following sections:

Configuring the Input

Follow this example procedure to get a better view of the process and walkthrough of the errors you may encounter. It uses one of our well-known sample datasets, the `sales_data.csv` file which is located in `<PDI_ROOT>\samples\transformations\files`.

1. In Spoon, begin with a **Text file input** step.
2. Add a **Text file output** step.
3. Connect the steps with a single hop:

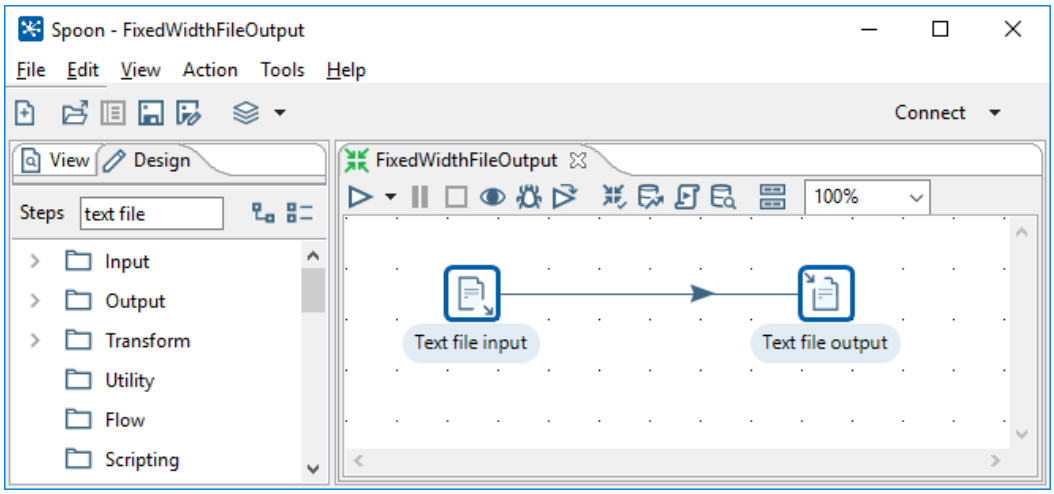
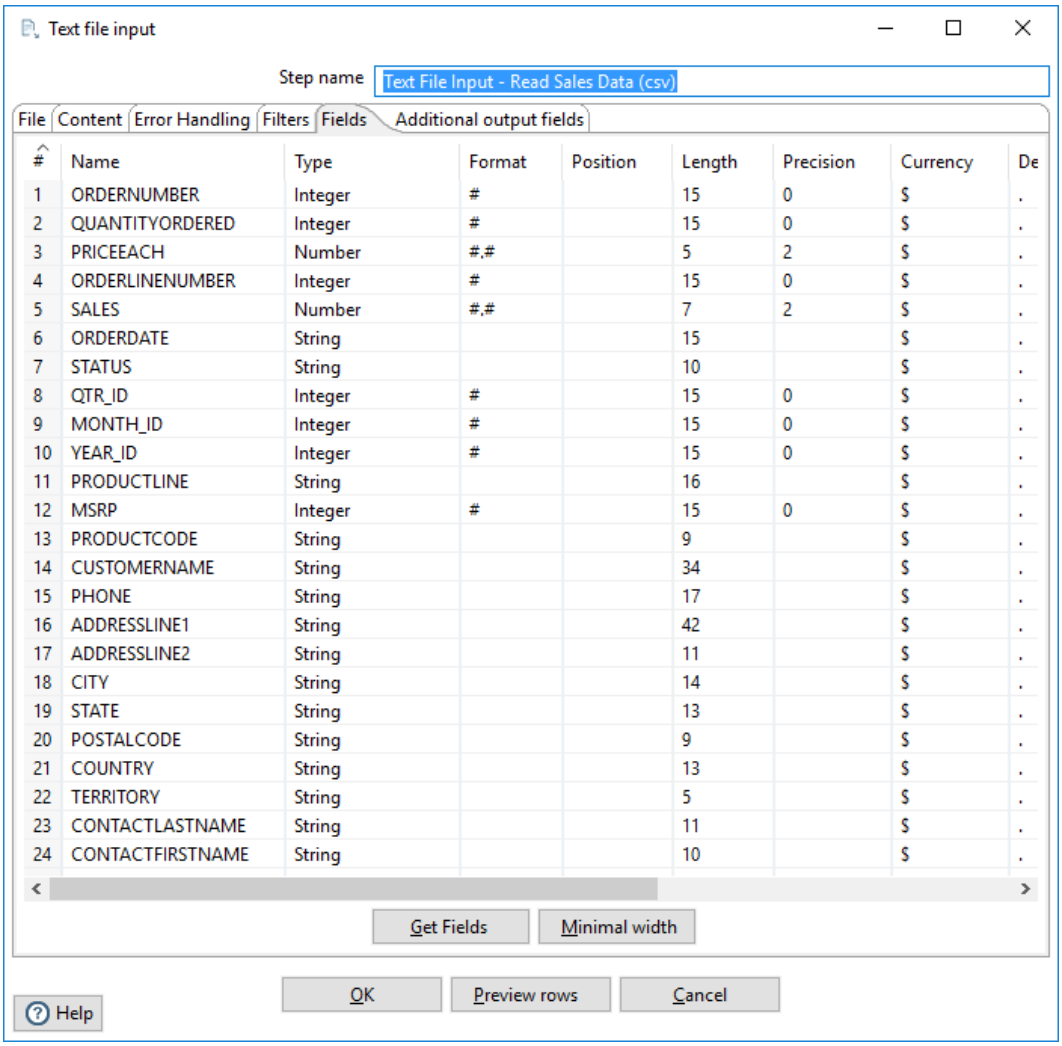


Figure 1: Text File Input / Output Steps

4. Double-click on **Text file input** to open the step configuration.
5. On the **File** tab, rename the step to something more descriptive, such as **Text File Input – Read Sales Data (CSV)**.
6. Point the step to a single file: `sales_data.csv`.
7. Copy it into your sample's directory to gain access to `${Internal.Transformation.Filename.Directory}\sales_data.csv`, for example.
8. On the **Content** tab, change the separator from a semicolon to a comma.
9. Click on the **Fields** tab, and then select **Get Fields** to read the header information.
10. Because the file you are working with is small, set the number of fields to zero when you are prompted. This will sample all the lines for the file.

11. Click on **OK**, and you should end up with something similar to Figure 2:



#	Name	Type	Format	Position	Length	Precision	Currency	De
1	ORDERNUMBER	Integer	#		15	0	$	.
2	QUANTITYORDERED	Integer	#		15	0	$	.
3	PRICEEACH	Number	#.#		5	2	$	.
4	ORDERLINENUMBER	Integer	#		15	0	$	.
5	SALES	Number	#.#		7	2	$	.
6	ORDERDATE	String			15		$	.
7	STATUS	String			10		$	.
8	QTR_ID	Integer	#		15	0	$	.
9	MONTH_ID	Integer	#		15	0	$	.
10	YEAR_ID	Integer	#		15	0	$	.
11	PRODUCTLINE	String			16		$	.
12	MSRP	Integer	#		15	0	$	.
13	PRODUCTCODE	String			9		$	.
14	CUSTOMERNAME	String			34		$	.
15	PHONE	String			17		$	.
16	ADDRESSLINE1	String			42		$	.
17	ADDRESSLINE2	String			11		$	.
18	CITY	String			14		$	.
19	STATE	String			13		$	.
20	POSTALCODE	String			9		$	.
21	COUNTRY	String			13		$	.
22	TERRITORY	String			5		$	.
23	CONTACTLASTNAME	String			11		$	.
24	CONTACTFIRSTNAME	String			10		$	.

Figure 2: Text File Input - Get Fields Option

Configuring the Output

Next, configure the **Text file output** step to try to save this as fixed-width text.

1. Double-click on **Text file input** to open the step configuration.
2. On the **File** tab, rename the step to something more descriptive, such as **Text File Output - Save Fixed Width**.
3. For a filename, use the following value to save it in the transformation directory: `${Internal.Transformation.Filename.Directory}\sales_data_fixed_width`. Leave the extension field with the default `txt`.
4. On the **Content** tab, clear the fields named **Separator** and **Enclosure**, because fixed files contain neither attribute.
5. On the same tab, clear the **Header** checkbox, because headers can be problematic in a fixed-width file.

6. On the **Fields** tab, click on **Get Fields** to obtain the metadata information from the step that you configured earlier. It should look about the same as it did for the **Get Fields** in Figure 2.
7. Save your transformation.
8. Run the transformation and see if it saves fixed-width data.

Testing the Output

Although you are able to load the output file `sales_data_fixed_width.txt` into a text editor, you will encounter the problem shown below:

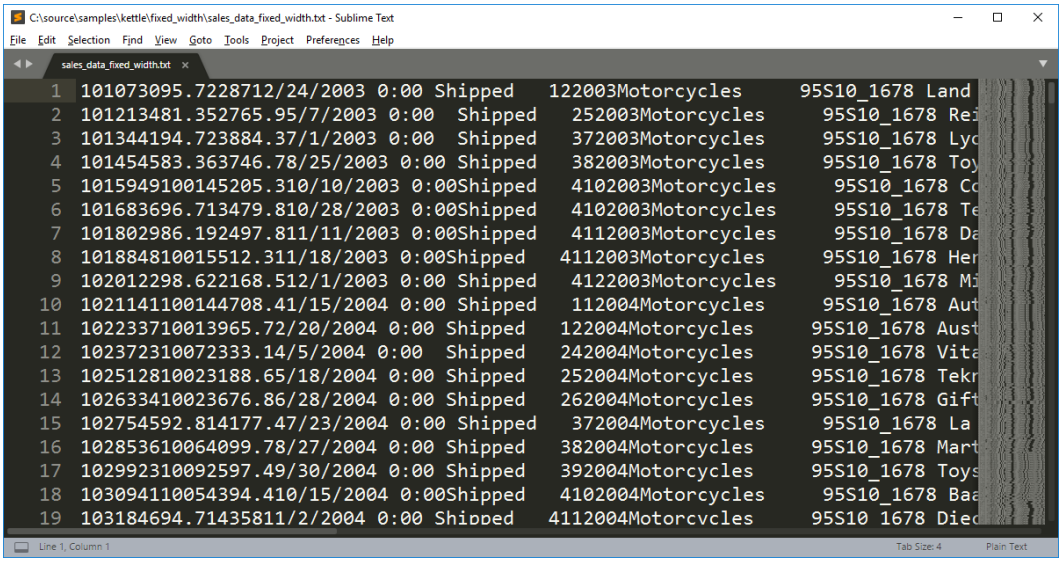


Figure 3: Testing the Output - Error

Your output will be jagged, while a properly formatted fixed-width file should have all its fields lined up vertically. If you go to the end of a line and press the down arrow a few times, you can determine the problem by focusing on the column that the editor says you are on for each line. You will see different results for each line, if you scroll down:

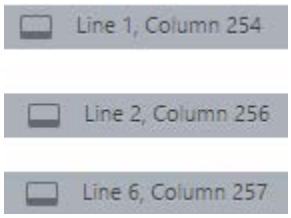
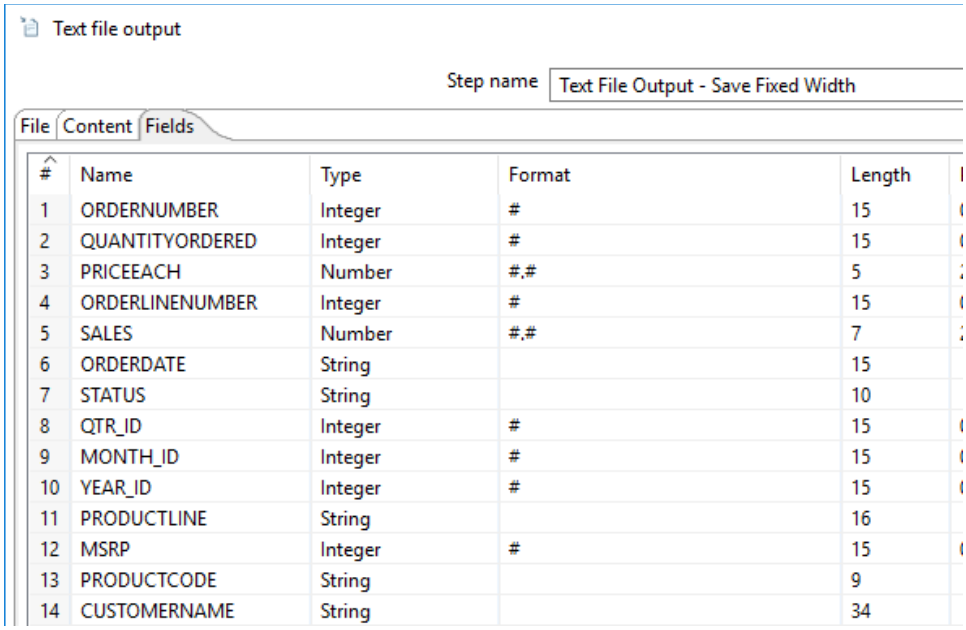


Figure 4: Varying Column Numbers

Having variable width fields and having lines of different lengths are symptoms of the same problem. We will fix that problem in the next section.

Fixing the Jagged Column Problem

The jagged column problem occurs because when taking the typical actions of telling PDI to write the file, and trusting the data types we received from the metadata in the **Text file input** step. In most cases, that would be the right way to go about things, but in this case, when the Text file output step is writing the file, it prioritizes the data type and format over the length. As you can see from the **Fields** tab on the **Text file output** step, there are many fields of the types **Number** and **Integer**:

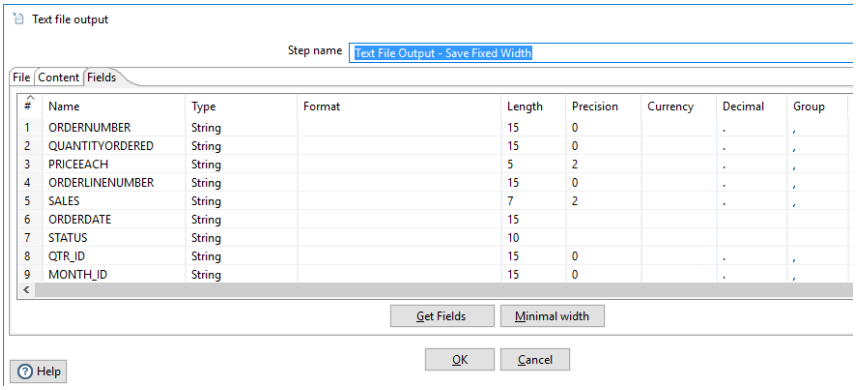


#	Name	Type	Format	Length
1	ORDERNUMBER	Integer	#	15
2	QUANTITYORDERED	Integer	#	15
3	PRICEEACH	Number	#,#	5
4	ORDERLINENUMBER	Integer	#	15
5	SALES	Number	#,#	7
6	ORDERDATE	String		15
7	STATUS	String		10
8	QTR_ID	Integer	#	15
9	MONTH_ID	Integer	#	15
10	YEAR_ID	Integer	#	15
11	PRODUCTLINE	String		16
12	MSRP	Integer	#	15
13	PRODUCTCODE	String		9
14	CUSTOMERNAME	String		34

Figure 5: Mismatched Field Types

These mismatched field types cause the columns to have irregular lengths, which creates the jagged appearance you saw in Figure 3. Here is a way to fix this problem:

1. Set all the fields with types **Integer** or **Number** to **String**.
2. Clear the **Format** and **Currency** fields, as shown below:



#	Name	Type	Format	Length	Precision	Currency	Decimal	Group
1	ORDERNUMBER	String		15	0		.	,
2	QUANTITYORDERED	String		15	0		.	,
3	PRICEEACH	String		5	2		.	,
4	ORDERLINENUMBER	String		15	0		.	,
5	SALES	String		7	2		.	,
6	ORDERDATE	String		15				
7	STATUS	String		10				
8	QTR_ID	String		15	0		.	,
9	MONTH_ID	String		15	0		.	,

Figure 6: Clear Format and Currency Fields

*You may also want to clear the **Decimal** and **Group** fields, but it is not necessary for our purposes.*

Examining the Data

When you reopen the file in a text editor, you will see that the format has changed, and the fields now line up in fixed-width format:

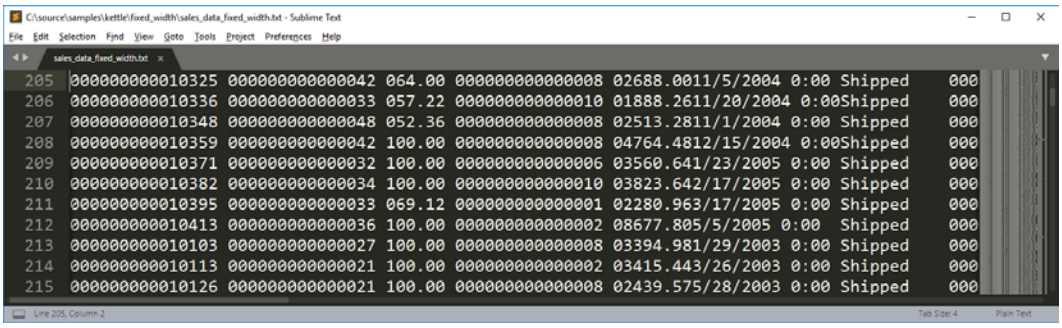


Figure 7: Fixed-Width File

Notice that the dates ending in 0:00 appear to be jagged at first glance, but this is because they are left-justified and vary in width; for example, in line `205`, `02688.0011/5/2004` is shorter than the date in line `206`, `01888.2611/20/2004`. The total size of the `date` column is fixed, as you can see from the fact that the column reading `Shipped` still appears to be even.

*Optionally, clean up this date column appearance in PDI by setting the trim column to **right** or **both**.*

There are spaces around some of the data, and the data marked as **numeric** by the previous **Text file input** step is padded with leading zeros. The size of the file can be further reduced by manipulating the field metadata (data type, length, etc.) of the source step. However, you need to thoroughly understand your source data to do this this safely and without truncation.

Reading Fixed-Width Data

Although we had to configure a general purpose output step to save fixed-width data, when reading fixed-width data, we have a specialized step available, **Fixed file input**.

Either manually configure the field size and type, or use **Get Fields** to visually select the field endpoints for the fixed-width file you sample:

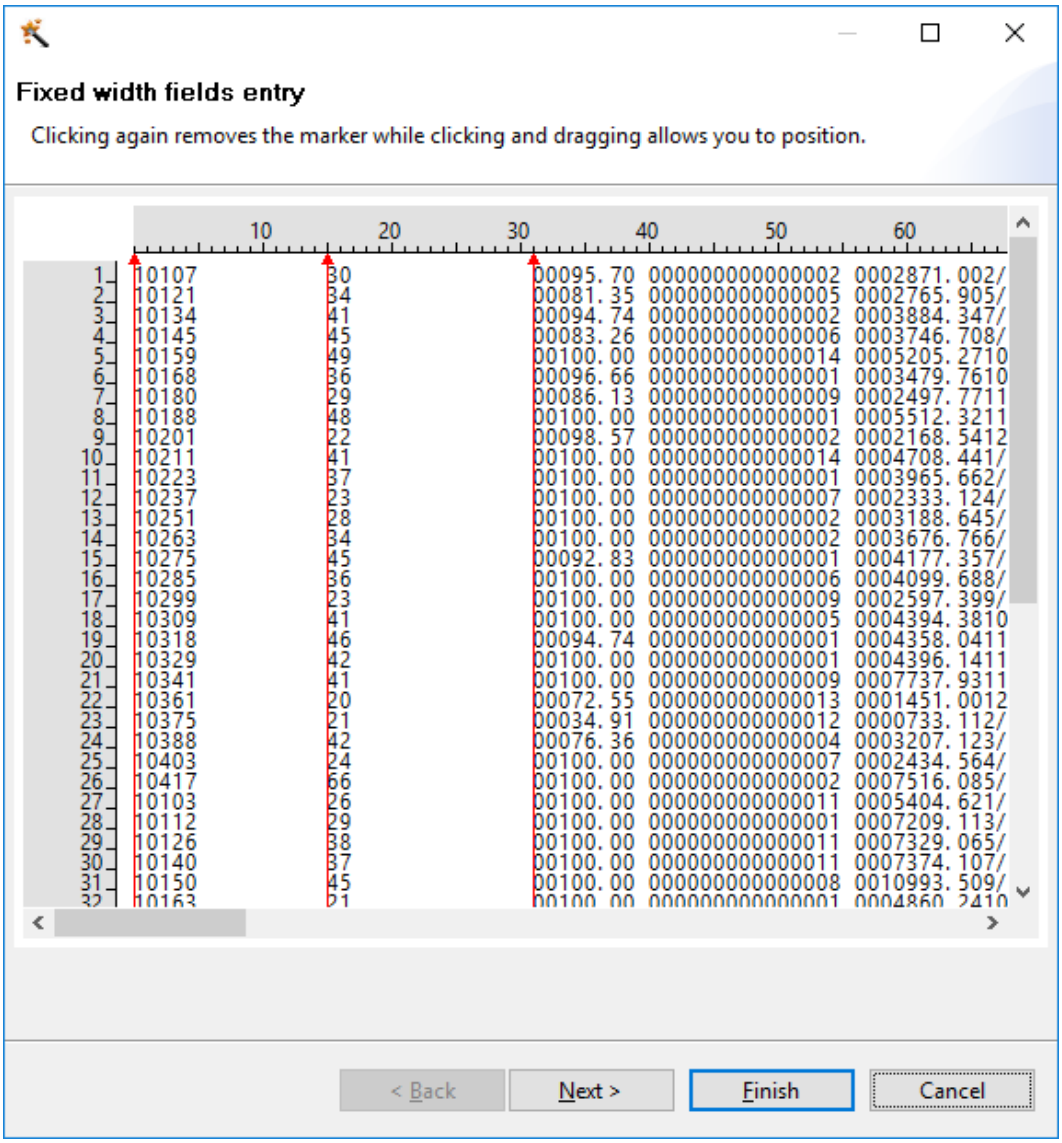


Figure 8: Fields Entry

Once you have visually selected the field widths, you may want to give your fields more descriptive names, because working through this view will save the field names as **Field1** and **Field2**, for example.

If you are working through a specification for the file, it may be easier to enter the names along with the width information directly in the fields grid.

Related Information

Here are some links to information that you may find helpful while using this best practices document:

- Pentaho Documentation
  - Components Reference
  - PDI Wiki

Finalization Checklist

This checklist is designed to be added to any implemented project that uses this collection of best practices, to verify that all items have been considered and reviews have been performed.

Name of the Project:_____

Date of the Review:_____

Name of the Reviewer:_____

Item	Response	Comments
Did you create a transformation to save fixed-width data?	YES_____ NO_____	
Did you test your transformation?	YES_____ NO_____	
Did you change field types as necessary?	YES_____ NO_____	
Did you use Fixed file input to read fixed-width data?	YES_____ NO_____	